J. Symbolic Computation (1990) **9**, 591–600

Computer Condensation of Modular Representations

A. J. E. RYBA

Department of Mathematics, University of Michigan, Ann Arbor, MI 48109, USA

(*Received* 15 *February* 1989)

Condensation replaces an explicit matrix representation of a group by a related but much smaller representation of a Hecke algebra. We give a general account of condensation and show how condensation is used to obtain the structure of large matrix representations. We include a detailed description of a new implementation of condensation. As an example of condensation, we work through the stages of the computation of the 2-modular character table of $G_2(3)$.

1. Introduction

The fundamental problem in representation theory is to decompose a matrix representation into its irreducible constituents. Parker introduced an extremely powerful computer program [the MEAT-AXE, see Parker (1984)] which solves this problem for matrices of dimensions up to around 1000. The aim of condensation is to preprocess the input data for the MEAT-AXE so that very much larger matrix representations can be decomposed. This paper includes a complete theoretical justification of condensation and a detailed description of a new condensation program. The new condensation program can be applied to a large class of representations in dimensions up to around 10^6. As an application of condensation, we calculate the 2-modular character table of $G_2(3)$ which was previously unknown.

Condensation programs allow us to use Parker's MEAT-AXE [see Parker (1984)] to study modules which would be much too large to investigage with the MEAT-AXE alone. A typical application of condensation consists of the following stages which we elaborate on in the subsequent sections of this paper.

(1) Select a group G and a kG-module V which is to be studied (k is assumed to be a finite field of characteristic p).
(2) Select a p'-subgroup H of G. The subgroup H gives rise to an idempotent

$$e = \frac{1}{|H|} \sum_{h \in H} h$$

of kG and thus to a Hecke algebra $ekGe$ (a subalgebra of kG).
(3) Apply condensation, which produces a representation of $ekGe$ on a "condensed" module $\tilde{V}$. $\tilde{V}$ is constructed as the subspace Ve of V (see sections 2–4).
(4) Use Parker's MEAT-AXE to analyse the $ekGe$-module $\tilde{V}$ [see Parker (1984)].
(5) Use the correspondence between kG-modules and $ekGe$-modules to obtain information about irreducible constituents and composition series of V (see Section 2).

0747–7171/90/050591+10 $03.00/0

Throughout this paper we shall use G, k, V, p, H, e, kG, $ekGe$ and $\tilde{V}$ as above without further comment. The key to the use of condensation is the relationship between kG-modules and $ekGe$-modules, and this relationship is explained in section 2. There are a number of rather different computer programs which carry out the third stage of the procedure described above, and we shall refer to these programs as condensation programs.

Before we look closely at the details of any condensation program it is worthwhile to consider exactly what sort of kG-modules we would like to condense. To begin with, in a useful application of condensation, $dim(V) \geq \mu$ (where μ is the dimension of the largest module which can be analysed with the MEAT-AXE), since for a smaller module the MEAT-AXE can be applied directly to V. To within an order of magnitude, μ is given by the dimension of the largest square matrix that can be stored in a computer's memory (typically μ is bounded above by 1000). Thus, in any condensation program, the module V is assumed to be so large that it cannot simply be specified by explicit matrices giving the actions of generators of G. However, in order to obtain an explicit matrix representation of $ekGe$ on $\tilde{V}$, it is necessary for a condensation program to be able to calculate the image of any given vector of V under any given element of G. Accordingly, condensation can only be applied to special kG-modules for which the group action can be specified by a compact formula. For example, there is a program which condenses permutation modules and another program which condenses exterior powers of small matrix representations. We note that a different condensation program is needed for each special type of kG–module.

The first condensation programs were written by Parker and Thackray in 1979. These programs were individually designed for use on particular modules of dimensions up to 15400 for the groups J_4, McL and Co_3. Amongst other applications, these condensation programs played an important role in the construction of J_4. Recently, two new condensation programs have been developed; these programs can be applied to a wide range of modules with degrees up to around 10^6. In this paper we shall concentrate on one of the new programs (written by Ryba in 1987) which condenses exterior powers of matrix representations. The other new program condenses permutation modules and it was written by Parker as a part of his most recent MEAT-AXE package. Although we examine only one condensation program in detail, much of our discussion also applies to other condensation programs.

2. Hecke Algebras and Their Modules

In this section we describe the Hecke algebras which we shall use and we collect a few well-known but useful results which relate their representations to those of G.

From the kG-module, V, we obtain an $ekGe$-module $\tilde{V} = Ve$ ($ekGe$ acts on the right). We say that $\tilde{V}$ is condensed from V. Since $\tilde{V}$ consists of the fixed points of the action of H on V, it is probably considerably smaller than V. Typically, we would expect $dim\tilde{V} \approx dimV/|H|$ and therefore it should be much easier to apply the MEAT-AXE to $\tilde{V}$ rather than to V. Moreover, any information about $\tilde{V}$ which we obtain with the MEAT-AXE gives rise to information about V via the following proposition.

PROPOSITION 1.

(i) *Let $X_1, X_2, \ldots, X_r$ be the irreducible constituents (composition factors) of V, then the irreducible constitutents of $\tilde{V}$ are the non-zero members of the set $\{\tilde{X}_1, \tilde{X}_2, \ldots, \tilde{X}_r\}$.*

(*ii*) *Let* $0 = V_0 \subset V_1 \subset \ldots \subset V_n = V$ *be a composition series of* V. *Then there is a sequence* $0 \leq i_0 \leq i_1 \leq \ldots \leq i_m \leq n$ *such that* $0 = \tilde{V}_{i_0} \subset \tilde{V}_{i_1} \subset \ldots \subset \tilde{V}_{i_m} = \tilde{V}$ *is a composition series of* $\tilde{V}$. *Moreover, a typical composition factor* $\tilde{V}_{i_k}/\tilde{V}_{i_{k-1}}$ *is isomorphic to the condensed module* $(V_{i_k}/V_{i_{k-1}})e$.

(*iii*) *Conversely, let* $0 = W_0 \subset W_1 \subset \ldots \subset W_m = \tilde{V}$ *be a composition series of* $\tilde{V}$. *Then* V *has a composition series* $0 = V_{01} \subset \ldots V_{0r_0} \subset V_{11} \subset \ldots V_{1r_1} \subset \ldots V_{m1} \subset \ldots V_{mr_m} = V$ *such that* $\tilde{V}_{ij} \cong W_i$.

The proposition is proved by combining the following lemmas. The proofs of the first two lemmas are very straightforward and are omitted. Our treatment of the other lemmas is similar to that given in Herstein (1968).

LEMMA 1. *If* $W \leq V$ *as* kG*-modules, then* $We \leq Ve$ *as* $ekGe$*-modules.*

LEMMA 2. *If* $W \leq V$ *as* kG*-modules, then* $(V/W)e \cong Ve/We$ (*an isomorphism of* $ekGe$*-modules*).

LEMMA 3. *If* $X \leq Ve$ *as* $ekGe$*-modules, then* $X = We$ *for some* kG*-submodule* W *of* V.

PROOF. Let W be the kG-submodule of V given by the vectors of XkG. Noting that e acts as an identity on Ve we obtain: $X = Xe \leq XkGe = We = XekGe \leq X$. Thus $X = We$.

LEMMA 4. *If* V *is an irreducible* kG*-module, then* Ve *is either irreducible or zero when regarded as an* $ekGe$*-module.*

PROOF. Otherwise, we would have a submodule X with $0 < X < Ve$. Then $X = We$ for some proper submodule W of V (by Lemma 3), and this contradicts the irreducibility of V.

We shall now illustrate a fairly typical application of Proposition 1. This application arose as the final phase of the calculation of the 2-modular character table of $G_2(3)$ [which was needed in Ryba *et al.* (1989)]. Many similar problems arise in the computation of other modular character tables.

All but one of the irreducible 2-modular characters of $G_2(3)$ are easily determined (from character theoretic information) by the MOC3 computer system of Parker, Lux & Hiss. Indeed, the MOC3 system leads to the following information about the principal block of the decomposition matrix (the non-principal blocks all have defect zero and therefore cause no problems).

	Ordinary irreducibles (ATLAS notation)																	
	1	14	78	91*a*	91*b*	91*c*	104	168	182*a*	182*b*	273*a*	273*b*	546*a*	546*b*	728*a*	728*b*	729	819
1	1			1	1	1					1	1					1	1
14		1					1		1	1	1	1			1	1	1	1
78			1					1	1	1	1	1	y	y	$1+y$	$1+y$	$1+y$	$1+y$
90*a*				1			1	1			1	1	x	x	x	x	x	$1+x$
90*b*					1					1		1	1			2	1	1
90*c*						1			1		1			1	2		1	1
χ													1	1	1	1	1	1

The left-hand column contains the 2-modular irreducible characters (including the unknown character χ). Certain entries in the table depend on a pair of undetermined positive integers which we denote by x and y. In order to pin down the values of x and y we need to find the irreducible constituents of a module whose character definitely contains χ. A convenient module is obtained as a 2-modular reduction of the 1001-dimensional module $\Lambda^4 14$ whose (ordinary) character is $91b+91c+819$. We shall also denote the 2-modular reduction by $\Lambda^4 14$ and, although this module could be defined over any field of characteristic 2, we shall work over the field $k = F_{64}$ (this choice of field is explained in section 5). From our decomposition matrix we deduce that the modular character of $\Lambda^4 14$ is

$$1^3+14+78^{1+y}+90a^{1+x}+90b^2+90c^2+\chi.$$

In section 5, we give a detailed description of the condensation of $\Lambda^4 14$ with respect to a Frobenius subgroup of order 21 in G. In particular, we obtain the following table of condensed modules in section 5.

kG-modules	1	14	78	$90a$	$90b$	$90c$	$\Lambda^4 14$
condensed $ekGe$-modules	1	0	8	4	$2a$	$2b$	$1^3+8^2+4+2a^2+2b^2+14$

The second row of the table gives the irreducible constituents of the condensed modules, and these irreducible constituents are named by their degrees together with an appropriate suffix if necessary. Observe that the 14-dimensional kG-module actually condenses to the zero module for $ekGe$. We also note that the 14-dimensional irreducible module for $ekGe$ must be the module obtained by condensing the unknown irreducible representation of $G_2(3)$.

We can now apply Proposition 1(i) to deduce that the irreducible constituents of $\Lambda^4 14$ are $1^3+14^z+78^2+90a+90b^2+90c^2+\chi$, where z is an undetermined positive integer. Comparing this with our earlier expression for the character of this module, we deduce that $x=0$, $y=1$ and thus χ is an irreducible character of degree 378.

We observe that in this application only the first part of Proposition 1 is used. It turns out that the other parts of the proposition are needed to determine the constituents of $\Lambda^4 14$ in Section 5.

3. Computer Representation of Representations

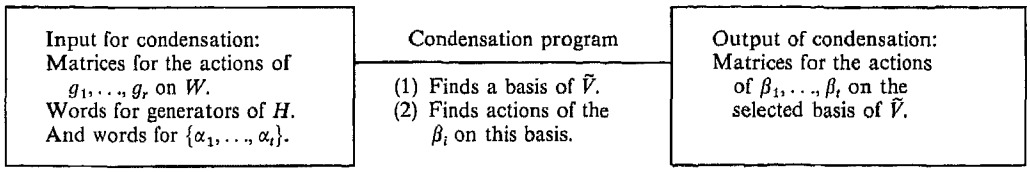
In this short section we shall concentrate on the nature of input and output data for a condensation program. We shall restrict our attention to the particular program which condenses a module of the form $V \cong \Lambda^n W$. In order to create data representations of modules for kG and $ekGe$, the user of the condensation program must solve the following three preliminary problems.

PROBLEM 1. Identify a set of generators $g_1, g_2, \ldots, g_r$ for G and obtain matrices giving their actions on a fixed basis of W. The actions of $g_1, \ldots, g_r$ completely determine the kG-module W.

PROBLEM 2. Give words in $g_1, g_2, \ldots, g_r$ for generators $h_1, h_2, \ldots, h_s$ of the subgroup $H \leq G$.

PROBLEM 3. Give words in $g_1, g_2, \ldots, g_r$ for a set of group elements $\{\alpha_1, \alpha_2, \ldots, \alpha_t\}$ such that $\{\beta_1 = e\alpha_1 e, \beta_2 = e\alpha_2 e, \ldots, \beta_t = e\alpha_t e\}$ generate $ekGe$. Modules for the Hecke algebra will be described by matrices giving the actions of $\beta_1, \ldots, \beta_t$.

The solutions to Problems 1 and 2 are used to organise the input of the condensation, while Problem 3 determines exactly what output is produced. The condensation program must perform the following pair of functions:



We observe that the explicit matrix representation of *ekGe* which is produced as output by the condensation program is suitable as direct input for Parker's MEAT-AXE [see Parker (1984)].

Problems 1 and 2 are familiar to users of Parker's MEAT-AXE and they are easily solved. Unfortunately, although it is easy to write down random sets of group elements which probably solve Problem 3, there is no known computational procedure to verify that a given set of elements do generate *ekGe*. In practice, the user can bypass Problem 3 since a condensation program will run and produce useful output regardless of whether $\beta_1, \ldots, \beta_t$ generate the full Hecke algebra. In section 5 we illustrate one way of avoiding the problem of generation of *ekGe*.

4. A Condensation Algorithm

In this section we shall describe a computer program which condenses a kG-module of the form $V \cong \Lambda^n W$ with respect to a subgroup $H \leq G$. The program is intended for use on modules with $\dim(V) \leq 10^6$. Since we shall feed the output of our program into Parker's MEAT-AXE, we shall also suppose that $\dim(\tilde{V}) \approx \dim(V)/|H| \leq 1000$. One extra limitation of our algorithm is that the subgroup H must act monomially on the space W (in other words, W must have a basis whose members are permuted and possibly rescaled by the action of H). An important benefit gained from this restriction is that the time taken by our condensation program decreases as the size of H increases. Usually, it is easy to find subgroups which act monomially on particular representations and so the added requirement on H does not present any great difficulty.

As we observed in section 3, the condensation program has to perform two functions. It is convenient to divide up these tasks into a pair of programs which we shall call precondensation and matrix condensation. Precondensation is used to calculate an appropriate basis for $\tilde{V}$ while matrix condensation calculates the action of a typical element $e\alpha e \in ekGe$ on the selected basis of $\tilde{V}$ (this will be repeated for all elements $e\alpha e$ in a generating set for *ekGe*).

The precondensation and matrix condensation programs turn out to consist of some rather easy calculations in permutation group theory and linear algebra, respectively. In order to describe the programs we must now introduce some notation and appropriate co-ordinates for W and V.

Let $\{w_1, \ldots, w_m\}$ be a basis which exhibits the monomial action of H on W. We index the members of this basis by $S = \{1, \ldots, m\}$. From the monomial action of H we obtain a permutation representation of H on S. We denote this permutation action by right multiplication. Let F be the kernel of the permutation representation, thus F acts diagonally on our basis of W.

Now let i_1 and h denote typical elements of S and H so that $w_{i_1}h = \chi_{i_1}(h)w_{i_1h}$ for some function $\chi_{i_1}: H \to k$. Although χ_{i_1} is not necessarily a character of H, it does restrict to a character of F.

Let S^n denote the collection of unordered n-tuples of elements of S. Given $i \in S^n$, we order the members of i as $i_1 \leq \ldots \leq i_n$ and we obtain a vector $v_i = w_{i_1} \wedge \ldots \wedge w_{i_n} \in V$. Moreover, as i varies through S^n the corresponding vectors v_i form a basis of V.

To describe a basis of $\tilde{V}$ we consider the set, $\mathscr{I}$, of orbits of H on S^n. The computation of $\mathscr{I}$ could be a rather long process, but as we shall see later it is possible to compute a basis of $\tilde{V}$ without obtaining all of these orbits. For each orbit $I \in \mathscr{I}$ we select a representative $i \in I$ and let

$$v_I = v_i \sum_{h \in H} h.$$

Thus v_I is a linear combination $\sum_{j \in I} c_j v_j$, and the non-zero vectors of this form give the required basis of $\tilde{V}$. Let $\mathscr{I}^* = \{I \in \mathscr{I} | v_I \neq 0\}$. This set indexes our basis of $\tilde{V}$. We also introduce

$$S^{n*} = \bigcup_{I \in \mathscr{I}^*} I \subset S^n.$$

$\mathscr{I}^*$ arises as the set of orbits of H on S^{n*}.

In an actual computation of $\mathscr{I}^*$ we would like to avoid the calculation of the orbits of H on S^n. The following observations show that we can actually get away with a much smaller orbit calculation. We first note that if $i = \{i_1, \ldots, i_n\} \in S^{n*}$, then $\chi_{i_1}\chi_{i_2} \ldots \chi_{i_n}$ must restrict to the trivial character of any subgroup $E \leq F$ (for otherwise, there is an $f \in F$ with $\chi_{i_1}(f)\chi_{i_2}(f) \ldots \chi_{i_n}(f) = \lambda \neq 1$, giving

$$v_I = v_i \sum_{h \in H} h = v_i \sum_{h \in H} fh = \lambda v_I,$$

and therefore $v_I = 0$). Accordingly, let S_E^{n*} denote the subset of S^n consisting of those $i = \{i_1, \ldots, i_n\}$ for which $\chi_{i_1}\chi_{i_2} \ldots \chi_{i_n}$ is trivial on E. Let $\mathscr{I}_E^*$ be the set of those orbits of H on S^n which are entirely contained within S_E^{n*}. Then

$$S^{n*} \subset S_F^{n*} \subset S_E^{n*} \subset S_1^{n*} = S^n \quad \text{and} \quad \mathscr{I}^* \subset \mathscr{I}_F^* \subset \mathscr{I}_E^* \subset \mathscr{I}_1^* = \mathscr{I}.$$

It follows that we can always compute $\mathscr{I}^*$ by searching through any convenient set of the form $\mathscr{I}_E^*$.

By summarising the constructions of $\mathscr{I}^*$ and S^{n*} we reduce the precondensation program to the following sequence of standard steps.

STEPS OF THE PRECONDENSATION PROGRAM

(1) Calculate a basis of W on which H acts monomially.
(2) Calculate a generating set for a convenient subgroup $E \leq F$.
(3) Calculate and store S_E^{n*}.
(4) Calculate $\mathscr{I}_E^*$ (this requires an orbit calculation on a set of size about $\dim(V)/|E|$).
(5) For each $I \in \mathscr{I}_E^*$, pick $i \in I$ and calculate

$$v_I = v_i \sum_{h \in H} h = \sum_{j \in I} c_j v_j.$$

The set $\mathscr{I}^*$ consists of those I for which all of the resulting coefficients c_j are non-zero. For each $i \in I$, store the triple (i, I, c_i).
(6) For all $I \in \mathscr{I}^*$ print out all triples (i, I, c_i).

In my present implementation of precondensation, the first two steps are the responsibility of the program user. They could be automated in a future implementation. In step (2), the user can supply generators for any convenient subgroup $E \leq F$, but the later steps of the program will run fastest when $E = F$. The output of the precondensation program is a list of approximately $dim(V)/|F|$ triples. The output list clearly contains exactly the information required to obtain all of the co-ordinates of the basis vectors v_I for $\tilde{V}$.

The matrix condensation program must calculate the IJ-entries of the matrix of $e\alpha e$ where I and J vary over $\mathscr{I}^*$. The program uses the output list of precondensation together with the matrix of the action of α on W as its data. The following lemma gives an efficient method for calculating the matrix of $e\alpha e$. In the lemma, i and j index basis vectors of $\Lambda^n W$ and they represent n-tuples.

LEMMA 5.

(*i*) *Suppose that*

$$v_i\alpha = \sum_j \alpha_{ij} v_j,$$

so that α_{ij} is the ij entry of the matrix of the action of α on V. Then α_{ij} is obtained from the matrix of the action of α on W by calculating the determinant of the $n \times n$ submatrix whose rows and columns are indexed by the (ordered) n elements of i and j respectively.

(*ii*)
$$v_I e\alpha e = \sum_J A_{IJ} v_J \quad \text{where} \quad A_{IJ} = \frac{1}{|J|} \sum_{\substack{i \in I \\ j \in J}} \frac{c_i}{c_j} \alpha_{i,j}.$$

PROOF. Part (i) is elementary. For part (ii) we calculate

$$v_I e\alpha e = v_I \alpha e = \frac{1}{|H|} \sum_{\substack{i \in I \\ h \in H}} c_i v_i \alpha h = \frac{1}{|H|} \sum_{J \in \mathscr{I}^*} \sum_{\substack{j \in J \\ i \in I \\ h \in H}} c_i \alpha_{ij} v_j h = \frac{1}{|H|} \sum_{J \in \mathscr{I}^*} \sum_{\substack{i \in I \\ j \in J}} c_i \alpha_{ij} \frac{|H|}{|J| c_j} v_J.$$

To obtain the last of these equalities, we set

$$v_j \sum_{h \in H} h = \lambda v_J = \lambda \sum_{k \in J} c_k v_k.$$

The coefficient of v_j on the left-hand side is $|H|/|J|$ (since if $h_0 \in H$ rescales v_j to μv_j, then

$$v_J = v_j \sum_{h \in H} h_0 h = \mu v_J$$

and thus $\mu = 1$). Therefore $\lambda = |H|/|J| c_j$.

The matrix condensation program consists of the following simple implementation of the calculation described in Lemma 5.

STEPS OF THE MATRIX CONDENSATION PROGRAM

(1) Initialise entries of the output matrix, A_{IJ}, to 0.
(2) For each pair of triples (i, I, c_i) and (j, J, c_j), calculate the $n \times n$ determinant $\alpha_{i,j}$ and add $(c_i/|J|c_j)\alpha_{i,j}$ onto the current value of A_{IJ}.
(3) Print out the matrix A.

The most time-consuming part of the whole condensation process is the repeated calculation of determinants needed in matrix condensation. If we let τ represent the time needed to compute one determinant, then each run of matrix condensation will last for a time of about $(dim(V)/|F|)^2\tau$. In an implementation of condensation where H is not assumed to be monomial and therefore a nice basis of W cannot be used, the corresponding time would be about $[dim(V)]^2\tau$.

The actual implementations of precondensation and matrix condensation contain one trick which is used to reduce the value of τ. Large blocks of common entries are shared by many of the different determinants whose values are needed. By lexicographically ordering the output list of precondensation, we make it likely that in the matrix condensation program successive determinants will share a large common block of entries. This allows us to avoid a great deal of repetition in the calculation of the determinants and dramatically reduces the value of τ.

5. An Example

In this section, we justify the table of condensed modules of $G = G_2(3)$ which was given in section 2. The subgroup which we use in the condensation process is a Frobenius group of order 21 and is denoted by H. Since H is a Frobenius group it acts monomially on any representation defined over the field $k = F_{64}$ of size 64. We let W be the 14-dimensional irreducible kG-module. Matrices generating the representation afforded by W are obtained as 2-modular reductions of the matrices for $G_2(3)$ specified in the Atlas (Conway *et al.*, 1985). Our aim is to compute and analyse the condensed modules obtained from the exterior powers of W.

Before we can run any programs, we must begin by choosing a new coordinate system for W so that H is represented by monomial matrices. These new coordinates of W will henceforth be fixed and whenever we refer to the matrix of an endomorphism of W, this new coordinate system will be implicitly assumed. We also need to locate a subgroup F which has order 7 and acts diagonally on our new basis.

The matrix generators for H and F constitute the only data needed by our precondensation program. In the following table, we summarise the performance of precondensation on modules $V \cong \Lambda^n W$. We include the number of cpu seconds consumed on a Sun 3/60 in each run of the program.

n	$dim\ V = \|S^n\|$	$dim\ \tilde{V} = \|\mathcal{S}^*\|$	$\|S^*\|$ = size of output list	cpu time
1	14	0	0	0·1 s
2	91	5	13	0·1 s
3	364	20	52	0·2 s
4	1001	45	135	0·4 s

The actual application of the matrix condensation program is also a very straightforward procedure. However, as we observed in section 3, in order to use matrix condensation, we need to produce a set of generators of the Hecke algebra $ekGe$. Any sufficiently large randomly selected subset of $ekGe$ is likely to be a set of generators, although it is very hard to prove generation. To get round this problem, we randomly choose a set of potential generators of $ekGe$ and we let R denote the (unknown) subalgebra that is really generated by our chosen elements. The matrix representation computed by condensation from a given kG-module is just the restriction to R of the actual condensed module.

We tabulate the results of condensations of exterior powers of W in the following table. The second row of the table is obtained from the partially calculated decomposition matrix displayed in section 2. The entries in the third row are obtained from those in the second row by applying Proposition 1 together with a character theoretic calculation of dimensions of fixed point spaces of H. The unknown *ekGe*-module obtained by condensing χ is denoted by $\tilde{\chi}$. The information given in the fourth row is obtained by applying condensation and analysing the resulting representations of R with the MEAT-AXE.

kG-module	$\Lambda^2 W$	$\Lambda^3 W$	$\Lambda^4 W$
kG-module decomposition	$1+90a$	14^2+78^2 $+90b+90c$	$1^3+14^1+78^{1+y}+90a^{1+x}$ $+90b^2+90c^2+\chi$
Decomposition of condensed Hecke algebra module	$1+4$	8^2 $+2a+2b$	$1^3+8^{1+y}+4^{1+x}$ $+2a^2+2b^2+\tilde{\chi}$
Decomposition of the R-restriction of the condensed module	$1+4$	8^2 $+2a+2b$	1^3+8^2+4 $+2a^2+2b^2+14$
Cpu (s)	0·3	1·3	10·2

When we compare the cpu timings with those used by the precondensation program, we see that, in this example, matrix condensation is the more expensive part of the condensation process.

It is clear from the decompositions of the modules condensed from $\Lambda^2 W$ and $\Lambda^3 W$ that the irreducible *ekGe*-modules which we have called, 1, 8, 4, $2a$ and $2b$ all restrict to irreducible R-modules. To complete the table of section 2 we need to show that $\tilde{\chi}$ also restricts to an irreducible R-module. From the decompositions of $\widetilde{\Lambda^4 W}$, we see that the restriction of $\tilde{\chi}$ to R is either 14 or $14+8$ and we must rule out the latter possibility.

A careful application of the MEAT-AXE shows that the exact structure of the R-module $\widetilde{\Lambda^4 W}$ is given by

$$\begin{array}{ccccc} & & 2a \oplus 2b & & \\ & & 1 & & 8 \\ 1 \oplus 4 \oplus & & & \oplus & 14 \\ & & 1 & & 8 \\ & & 2a \oplus 2b & & \end{array}$$

We deduce that if $\tilde{\chi}$ is not irreducible as an R-module, then as a module for the full Hecke algebra, $\widetilde{\Lambda^4 W}$ must have one of the following pair of structures:

$$\begin{array}{ccccc} & & 2a \oplus 2b & & \\ & & 1 & & 22 \\ 1 \oplus 4 \oplus & & & \oplus & \\ & & 1 & & 8 \\ & & 2a \oplus 2b & & \end{array} \quad \text{or} \quad \begin{array}{ccccc} & & 2a \oplus 2b & & \\ & & 1 & & 8 \\ 1 \oplus 4 \oplus & & & \oplus & \\ & & 1 & & 22 \\ & & 2a \oplus 2b & & \end{array}$$

Applying Propositions 1(ii) and (iii) we deduce that in every composition series of $\Lambda^4 W$, the composition factors χ and 78 occur in the same order. This statement contradicts the self-duality of $\Lambda^4 W$, and thus we are forced to conclude that $\tilde{\chi}$ is actually an irreducible R-module. This completes the justification of the table of condensed modules given in section 2.

This example showed that condensation can provide an easy way of obtaining the probable structure of a given matrix representation. Indeed, the structure of the R-module

$\widetilde{\Lambda^4 W}$ was obtained entirely by computer and, moreover, it is extremely likely that R is the full Hecke algebra. However, we needed an *ad hoc* argument to establish the structure of $\Lambda^4\widetilde{W}$ as an *ekGe*-module. Until a procedure can be found to solve Problem 3 of section 3 it is likely that results obtained by condensation will have to be verified by similar arguments.

6. Larger Examples

We close by tabulating the cpu times needed in some applications of condensation to larger modules. The table below illustrates that matrix condensation is the expensive part of the condensation process. The modules which appear in the table are all exterior powers of the form $V = \Lambda^n W$ and the subgroups H and F of G are as described in section 4. The final two columns of the table give the cpu seconds used on a Sun 3/60 per run of precondensation and matrix condensation, respectively.

G	$\lvert F\rvert$	$\lvert H\rvert$	$dim(W)$	n	$dim(V)$	$\lvert S^{n*}\rvert$	$dim(\tilde{V})$		
$G_2(3)$	7	21	14	5	2002	286	98	1·0 s	65·2 s
$G_2(3)$	7	21	14	6	3003	429	147	1·7 s	210·6 s
$G_2(3)$	7	21	14	7	3432	480	160	2·1 s	384·4 s
$G_2(3)$	7	21	14	8	3003	429	147	1·9 s	430·9 s
$G_2(3)$	7	21	14	9	2002	286	98	1·4 s	266·9 s
Ly	243	69984	111	3	221815	696	7	9·7 s	22·1 s
Ly	243	69984	111	4	5989005	25380	125	264·7 s	22658·3 s

References

Conway, J. H., Curis, R. T., Norton, S. P., Parker, R. A., Wilson, R. A. (1985). *An ATLAS of Finite Groups.* Oxford: Clarendon Press.

Herstein, I. N. (1968). *Noncommutative Rings.* Carus Mathematical Monographs.

Parker, R. A. (1984). The computer calculation of modular characters (the MEAT-AXE). In: *Computational Group Theory* (Atkinson, M. D., ed.), pp. 207–274. London: Academic Press.

Ryba, A. J. E., Smith, S. D., Yoshiara, S. (1990). Some projective modules determined by sporadic geometries. *J. Alg.*, in press.